International Journal of Innovative Computing, Information and Control
Volume **14**, Number **3**, June **2018**

ICIC International ©2018 ISSN 1349-4198
pp. **929–946**

AN EFFICIENT HYBRID FEATURE FOR EDGE-PRESERVING BASED ON BLOCK TRUNCATION CODING AND TREE-STRUCTURED VECTOR QUANTIZATION WITH EDGE ORIENTATION CLASSIFICATION OF BIT-PLANES

Ya-Pei Feng and Zhe-Ming Lu

School of Aeronautics and Astronautics
Zhejiang University
No. 38, Zheda Road, Hangzhou 310027, P. R. China
zheminglu@zju.edu.cn

Received December 2017; revised March 2018

Abstract. *In this paper, we propose a hybrid feature which merges Block Truncation Coding (BTC) and Tree-Structured Vector Quantization (TSVQ) with classification of edge orientation bit-planes. BTC is known for its high quality of reconstructed images, much lower computational complexity and low compression ratios. Vector Quantization (VQ), on the other hand, results in very high compression ratios but with high complexity. To solve these problems, we propose an improved approach for feature extraction based on bit-plane classification according to the Edge Orientation of the Bit-Plane (EOBP) templates and combine it with TSVQ. As different areas of the image have different activities and contrasts, we encode the active blocks containing edges with BTC and EOBP templates, and encode the inactive blocks having low intensity variations between its pixels with the VQ. Different from the original BTC and VQ based feature extraction, we extract the edge orientation information by applying the EOBP classification on the bit-plane image blocks, as a result that we classify image blocks into 28 kinds with different edges. Simulation results show that our proposed method performs better in image compression and image retrieval.*

Keywords: Block truncation coding, VQ, Feature extraction, Edge Orientation of the Bit-Plane (EOBP) templates

1. **Introduction.** With the rapid spread of the Internet and fast advancement of the global network, the amount of digital image data accessible to users has grown enormously. Image databases are becoming larger and more widespread, and there is a growing need for effective and efficient image retrieval systems. For CBIR [1], many researches have been devoted to extracting features to improve the retrieval accuracy, while most of the images are recorded in the storage device in compression format for reducing the storage space requirement, so there is a great need to employ a feature descriptor derived from the compressed data stream. In compressed domain, features can be extracted from the compressed data stream without decoding process.

Block Truncation Coding (BTC) [2] and Vector Quantization (VQ) [3] are two typical block-based image coding schemes that can be used not only to efficiently compress images but also to extract compressed-domain features, and thus several VQ based and BTC-based image retrieval methods have been proposed recently. As is known, BTC compression algorithms are simple and perform well when blocks contain edges or regions of large intensity variation. Even though the BTC based scheme achieves much lower computational complexity, the biggest drawback of the basic BTC based image-coding algorithm is that the bit rate is high and the edge has a ladder effect. As its lowest attainable bit rate is limited, it often leads to the blocking effect in homogeneous area in image. On the other hand, the VQ-based techniques use a multilevel quantizer that is much more effective at compressing the image, resulting in higher compression ratio. However, VQ-based methods only consider intra-block correlation and ignore visually important regions, which are edge information; hence, it suffers from edge degradation and has limited retrieval performance. Furthermore, it is a more complex scheme to implement due to the generation of the required codebook.

In 1984, Gray proposed a VQ-based encoding method to compress images [3]. In the method, images are divided into several blocks, and then the sub-blocks are further mapped to the most similar codeword in the VQ codebook. As a result, all of the pixels in the image block are replaced by one VQ index. After all the image blocks are compressed, a VQ index table can be obtained. On account of its advantages, VQ is applied to many research fields, such as index compression, image retrieval, and information hiding. As one of popular image compression techniques, VQ is also used in several image retrieval schemes combined with other schemes in the compression domain.

Although it is a simple technique, BTC [2] has played an important role in the history of digital image coding in the sense that many advanced coding techniques have been developed based on BTC. In BTC, images are firstly divided into non-overlapping blocks of pixels, and each image block is represented with two specific quantizers to maintain its mean value and standard deviation to the original image block. Thus, in the encoding process, two quantizers, i.e., the high and low quantizers, and a bit-plane image are generated. Decoding is the simple process of placing the appropriate reconstruction value at each pixel location as per the bit-plane. In [4], Qiu derived two image content description features, and he used Block Color Co-occurrence Matrix (BCCM) and the other Block Pattern Histogram (BPH) to compute the similarity measures of images for content-based image retrieval applications. [5] proposed an effective feature for color image retrieval based on two pattern co-occurrence matrices generated from the BTC compressed Y image and VQ compressed Cb and Cr images, respectively. [6] presents a new image feature descriptor derived from the Direct Binary Search Block Truncation Coding (DBSBTC) data-stream without requiring the decoding process. Three image feature descriptors, namely Color Autocorrellogram Feature (CAF), Legendre Chromaticity Moment Feature (LCMF), and Local Halftoning Pattern Feature (LHPF), are simply constructed from the DBSBTC min quantizer, max quantizer, and its corresponding bitmap image, respectively. [7] presents a new approach to index color images using the features extracted from the Error Diffusion BTC (EDBTC). Similar to the BTC scheme, EDBTC looks for a new representation (i.e., two quantizers and a bit-plane image) for reducing the storage requirement. The EDBTC bit-plane image is constructed by considering the quantized error which diffuses to the nearby pixels to compensate the overall brightness, and thus, this error diffusion scheme effectively removes the blocking effect and false.

Modifications have been made to the basic block truncation-coding algorithm to improve its performance, for example, the Absolute Moment BTC (AMBTC) [23], Error Diffusion BTC (EDBTC) [7] and Dot-Diffused BTC (DDBTC) [24]. The goal of AMBTC is to preserve the mean and first absolute central moment of image blocks. The EDBTC employs the error kernel to generate the representative bitmap image. As a result EDBTC produces two color quantizers and a bitmap image, which are further processed using VQ to generate the image feature. A new approach was proposed in [24] to derive features from the DDBTC which can diffuse the quantization error of the current processed pixel into its neighbouring pixels using the diffused matrix and class matrix concurrently to generate the bitmap image. The DDBTC also searches the minimum and maximum values in an image block as two representative quantization levels.

BTC is a halftoning based scheme with good image quality, while the drawback of BTC based algorithm is that the bit rate is high and the edge has a ladder effect. And VQ is an efficient and popular block based lossy image compression technique with a high compression ratio, while the VQ based methods suffer from edge degradation. Inspired by the Structured Local Binary Harr Pattern (SLBHP) [16], Kirsch edge detector [17], and the image retrieval based on histograms of EOPs and VQ indices [9], we propose a new classification scheme for the bit-plane of an image to reserve the edge information of an image. In this paper, we shall show another attractive feature based on BTC and VQ by exploring the edge orientation of the bit-plane. Apart from the two quantizers and the bit-plane image after BTC coding, the edge orientations of the bit-plane image blocks are also taken into account. The Contrast and Color Histogram based Feature (CCHF) of an image is generated from the BTC encoded data streams without decoding. And the Edge Orientation of Bit-Plane Feature (EOBPF) is extracted from the more detailed classification of the edge orientation information.

The structure of our paper is organized as follows. In Section 2, we give a brief overview of the principles of BTC and VQ based image compression techniques. In Section 3, we propose the edge orientation of bit-plane templates and its related feature extraction scheme. The performance of the proposed scheme and the comparison results are shown in Section 4 and the paper is concluded in Section 5.

2. **Related Works.**

BTC is a local binarization process and BTC can be applied for image compression that requires simple processes including the encoding and decoding phases. Previous researches aimed at combining the BTC with other image processing techniques into a hybrid approach, vector quantization for example. Therefore, we propose a hybrid feature which merges the BTC and VQ [15] with classification of edge orientation bit-planes. The following two subsections introduce BTC and BTC-VQ based schemes respectively.

2.1. **BTC based compression.**

Firstly, we give a brief introduction of the basic BTC based compression. In BTC, the input image is first divided into non-overlapping 4 × 4 blocks, and each image block is expressed as $x(i, j)$ for each element at the position (i, j), $0 \leq i < 4$, $0 \leq j < 4$. Firstly, we calculate the mean of the existing pixel in each block as $\overline{x}$ and the mean of the square of the pixel as $\overline{x^2}$ in Equations (1) and (2) respectively

$$\overline{x} = \frac{\sum_{i=0}^{3}\sum_{j=0}^{3} x(i, j)}{4 \times 4} \tag{1}$$

$$\overline{x^2} = \frac{\sum_{i=0}^{3}\sum_{j=0}^{3} [x(i, j)]^2}{4 \times 4} \tag{2}$$

Then the standard deviation of the pixels can be calculated as follows

$$\sigma = \sqrt{\overline{x^2} - (\overline{x})^2} \tag{3}$$

With the same size with the image blocks, a 4 × 4 bit-plane can be established by the use of the following formula in Equation (4):

$$bp(i, j) = \begin{cases} 1 & x(i, j) \geq \overline{x} \\ 0 & x(i, j) < \overline{x} \end{cases} \tag{4}$$

If a pixel is greater than or equal to the block mean, the corresponding pixel position of the bit-plane will have a value of 1; otherwise it will have a value of 0. Hence, the pixels in the 4 × 4 blocks are divided into two categories according to the bit-plane values. Then, two mean pixel values, one for the pixels greater than or equal to the block mean and the other for the pixels smaller than the block mean are also calculated. At the decoding stage, for each block, the pixel position whose corresponding location in the bit-plane has a value of 1 is replaced by the high mean and the pixel position whose corresponding location in the bit-plane has a value of 0 is replaced by the low mean. The high and low mean values are defined as M_0 and M_1 as follows:

$$M_0 = \overline{x} - \sigma\sqrt{\frac{q}{m-q}} \qquad M_1 = \overline{x} + \sigma\sqrt{\frac{m-q}{q}} \tag{5}$$

where q is defined as the number of $x(i,j)$ satisfying $x(i,j) \geq \overline{x}$. Thus, in the decoding phase, M_0 is assigned to "0" in the bit-plane, and M_1 is assigned to "1" in the bit-plane.

Before describing the construction process, we give the knowledge of VQ. It works by dividing a large set of vectors into groups. Each group is represented by its centroid point, as in k-means [10] and some other clustering algorithms. Over the past several decades, VQ has been applied to many fields such as image compression [5], image retrieval [11], data hiding [12], secret key generation [13] and distributed detection [14]. An N-level k-dimensional vector quantizer Q can be viewed as a mapping from the k-dimensional Euclidean space R^k to its finite subset $Y = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$, where Y is called codebook, and $\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{ik})^{\mathrm{T}}$, $i = 1, 2, \ldots, N$, are called codewords. Ideally, each codeword $\boldsymbol{y}_i$ should be the centroid of the corresponding cell V_i. Typically, VQ has three main stages: offline codebook generation, online encoding and decoding stages. Taking image compression for example, in the encoding stage, the input image is segmented into non-overlapping blocks (typical of size 4 × 4). Then each input block (vector) is compared with the codewords in a given codebook to find its best match codeword. The index of the best match codeword is used to encode the input image block. In the decoding stage, the receiver decodes each received VQ index from the codebook based on a simple look-up approach. Both the encoding and decoding stages use the same codebook.

Combined with the BTC and VQ, following is the BTC-VQ based feature extraction process.

Step 1. The input image is segmented into 4 × 4 non-overlapping blocks, each 4 × 4 block is expressed as $x(m,n)$, $0 \leq m < 4$, $0 \leq n < 4$, and each block is viewed as a vector.

Step 2. Perform the BTC compression on each image block, and we will obtain the high and low mean values M_0 and M_1, and the bit-plane matrix $bp(m,n)$.

Step 3. For the high mean M_0 and low mean M_1, scalar quantizers are generated. While for bit-planes $bp(m,n)$, vector quantizer is generated. Here the representative codebook as $\mathbf{C} = \{c_i | i = 0, 1, \ldots, N-1\}$ should be generated based on all the input vectors $bp(m,n)$ using the well-known LBG algorithm [15], where c_i is called codeword and N is the codebook size.

Step 3.1. Choose the $bp(m,n)$ as the input image for the vector quantizer, initiate a codebook C_0, calculate the Euclidean distance between the initial codeword and the input vector, choose the minimum value and regard it as the closest codeword in the codebook. Thus, each training vector records the index of its closest codeword in the codebook and then we divide the training set into N clusters.

Step 3.2. Each codeword in the current codebook is updated with the centroid of those training vectors so that the current codeword is the closest codeword of them. So, the newly generated codebook is used in the next iteration to minimize the overall average distortion in the codebook design process. The segmentation and updating processes are repeated until the overall average distortion between the last two consecutive iterations is not noticeably or a certain number of iterations are reached. In the encoding stage, for each k-dimensional input vector, the nearest neighbour codeword $\boldsymbol{c}_i = (\boldsymbol{c}_{i1}, \boldsymbol{c}_{i2}, \ldots, \boldsymbol{c}_{ik})$

in the codebook C, satisfying the following condition:

$$d(\boldsymbol{bp}, c_i) = \min_{0 \leq j \leq N-1} d(\boldsymbol{bp}, c_j) \tag{6}$$

where $d(\boldsymbol{bp}, c_j)$ is the distortion between the input vector $\boldsymbol{bp}$ and the codeword $\boldsymbol{c}_j$, which can be defined as follows:

$$d(\boldsymbol{bp}, \boldsymbol{c}_j) = \sum_{l=1}^{k} (bp_l - c_{jl})^2 \tag{7}$$

Step 4. The index i of the nearest neighbor codeword assigned to the input vector $\boldsymbol{bp}$ is transmitted over the channel to the decoder. The decoder has the same codebook as the encoder. In the decoding phase, for each index i, the decoder merely performs a simple table look-up operation to obtain $\boldsymbol{c}_i$ and then uses $\boldsymbol{c}_i$ to reconstruct the input vector $\boldsymbol{bp}$. Compression is achieved by transmitting or storing the index of a codeword rather than the codeword itself.

Step 5. Get the scalar index of the high mean M_0 and low mean M_1, and the vector index of the bit-plane $bp(m, n)$. The combination of the indices, which inherently consider the spatial information in image blocks, can be used for image compression and image retrieval.

2.2. Tree-structured VQ.

In general, the LBG-VQ [15] is known for its search complexity and as a result the Tree-Structured VQ (TSVQ) [25] codebook is designed allowing fast search. The encoder acts as a classification tree by taking an input vector and then classifies it according to the best reproduction of this vector. The main idea of TSVQ is to perform tree search on codebook entries rather than full search where codewords are selected by a sequence of binary decision. Tree-structured VQ reduces encoding complexity, and allows for progressive transmission. In this paper tree-structured product-codebook VQ is proposed to carry out low-complexity, low-memory storage and progressive transmission. TSVQ has been studied extensively in vector quantization from the perspective of data compression.

There are several differences between VQ and TSVQ, the processes of the codebook generation algorithm are listed as follows.

Firstly, calculate the center v of all the training vectors.

Secondly, get the vector v_0 which is the farthest from v, and get the vector v_1 which is the farthest from v_0. Then take (v_0, v_1) as the clustering center and do k-means [10] clustering with $k = 2$.

Thirdly, we get 2 centroids or prototypes within the entire training set. And we can get a boundary between the two clusters, which would be a straight line based on the nearest neighbor rule.

Next, the input data are assigned to the 2 centroids.

Then, for the data assigned to each centroid (called a group), apply the 2 centroids to each group separately. As a result, the initialization can be done by splitting the centroid into two parts.

Repeat the above steps until the number of binary tree levels meets the requirements, just as the example shown in Figure 1.

3. Proposed Scheme.

3.1. Proposed Edge Orientation of Bit-Plane (EOBP) templates.

As is known, the BTC based scheme achieves much lower computational complexity, and the biggest drawback of the basic BTC based image coding algorithm is that the bit rate is high and the edge has a ladder effect. As its lowest attainable bit rate is limited, it often

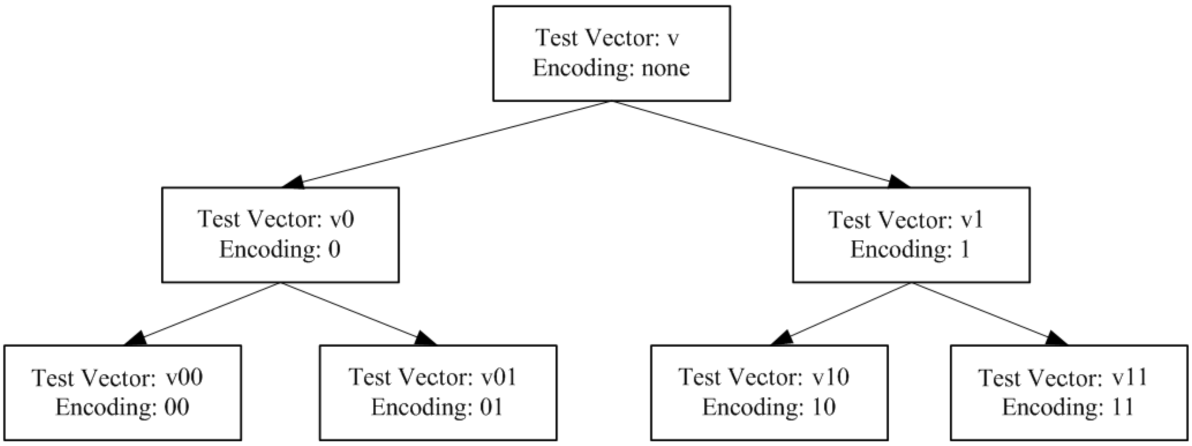


Figure 1. The basic structure of TSVQ

Original image

2	9	12	15
2	11	11	9
2	3	12	15
3	3	4	14

Bit-plane

0	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1

Reconstructed image

2	12	12	12
2	12	12	12
2	2	12	12
2	2	2	12

Figure 2. BTC compression image block

leads to the blocking effect in the homogeneous areas of the input image. The main characteristic of VQ is its high compression ratio. However, it is a more complex scheme during the codebook design process. Furthermore, VQ-based methods only consider intra-block correlation and ignore visually important regions, such as edge information; hence, it suffers from edge degradation and has limited retrieval performance. Therefore, we propose the modified scheme to overcome the disadvantages.

Just as we described in Section 2 for the BTC based compression, taking one of the image blocks for example, we will give a typical BTC compression process in Figure 2. According to Equations (1) and (3), calculate the mean value of the block as $\overline{x} = 7.94$, and the variance of the block as $\sigma = 4.91$. According to Equation (5), we can calculate the high and low means as $M_0 = 2.3$, $M_1 = 12.3$ respectively. Seen from the bit-plane in Figure 1, it is a binary distribution matrix consisting of '0' and '1'. And the distribution of '0' and '1' represents the edge orientation feature of the bit-plane. Although edges have only small populations in an image, the edges contain the most significant amount of orientation information. And there are many algorithms applying the edge orientation classifier to reserving the edge information of images.

Inspired by the Structured Local Binary Harr Pattern (SLBHP) [16], Kirsch edge detector [17], and the image retrieval based on histograms of EOPs and VQ indices [9], we explore the edge orientation distribution of the bit-plane. As the total number of possible bit-plane blocks with '0' and '1' is 2^{16}, while most of the edge orientation distributions are concentrated in the minority. The bit-plane of the image blocks varies from each other, and the edge orientations of the bit-plane describe their own characteristics such

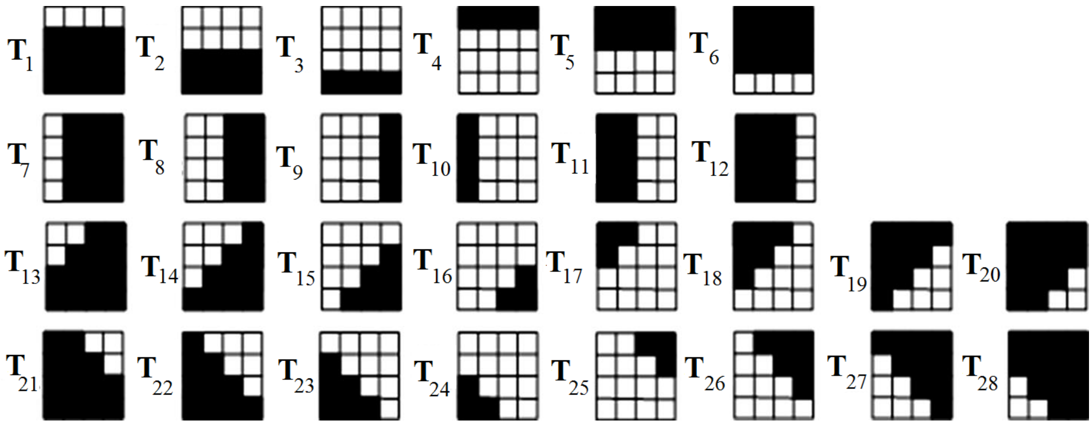


Figure 3. The 28 binary edge template classes

as horizontal edges, vertical edges, or diagonal edges. In this paper, a classifier based on the comparison of locally threshold image vectors with a predefined set of binary edge templates [20] is used. The predefined 28 binary edge templates ($\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_{28}$) are shown in Figure 3.

Here, according to the templates shown in Figure 3, we proposed a new Edge Orientation of Bit-Plane (EOBP) templates for the edge orientation classification of the bit-plane image. Therefore, we choose the representative matrix as our classification templates, and the twenty-eight 4 × 4 matrices for EOBP classification are shown in Figure 4.

In order to represent the edge orientations of the bit-plane, we define twenty-eight templates, i.e., $\mathbf{T}_1 \sim \mathbf{T}_{28}$ in Figure 4, which denote twenty-eight categories of edge orientations respectively and are corresponding to Figure 3. In addition, the detailed steps of the EOBP classification process are outlined below.

Firstly, when we get the bit-plane image $bp(m, n)$, we perform the twenty-eight 4 × 4 EOBP templates on each 4 × 4 block $bp(m, n)$, $0 \le m < 4$, $0 \le n < 4$. As a result, we will obtain an edge orientation vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_{28})$ with its components v_i $(1 \le i \le 28)$ being computed as follows:

$$v_i = \left| \sum_{m=0}^{3} \sum_{n=0}^{3} [bp(m, n) \oplus t_i(m, n)] \right| \quad 1 \le i \le 28 \tag{8}$$

where $t_i(m, n)$ stands for the element at the position (m, n) of the template $\mathbf{T}_i$.

Next, we calculate the maximum component $v_{\max}$ of its edge orientation vector $\boldsymbol{v}$, and then we choose the index i of the maximum value. As a result, we mark the block as the i-th category according to the following formula:

$$bp(m, n) \in \text{ The } i\text{-th edge category if } i = \underset{1 \le j \le 28}{\arg\max}\, v_j \tag{9}$$

From Equation (9), we can classify each bit-plane image block into one of twenty-eight categories according to its smoothness and its edge orientation. The $\mathbf{T}_1 \sim \mathbf{T}_{28}$ blocks represent different horizontal edges, vertical edges, or diagonal edges of the bit-plane image as shown in Figure 3.

Table 1 gives a specific example, which shows the number of training vectors in each category for the typical Lena and Baboon images of size 512 × 512. The total number of training vectors for each image is 16384, because the block size is 4 × 4 in this paper. In Table 1, indices 1 ∼ 28 stand for different edge orientations for bit-plane image.

$$\mathbf{T}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_5 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_6 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{T}_7 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_8 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \mathbf{T}_9 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{T}_{10} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_{11} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}, \mathbf{T}_{12} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix},$$

$$\mathbf{T}_{13} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_{14} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_{15} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_{16} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \mathbf{T}_{17} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_{18} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{T}_{19} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_{20} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}, \mathbf{T}_{21} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

$$\mathbf{T}_{22} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{T}_{23} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}, \mathbf{T}_{24} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix},$$

$$\mathbf{T}_{25} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_{26} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{T}_{27} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{T}_{28} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Figure 4. Twenty-eight templates

Table 1. The number of training vectors in each category for Lena and Baboon images

Category	1	2	3	4	5	6	7
Lena	945	1130	566	637	875	585	1075
Baboon	1382	1591	946	1014	1260	923	650
Category	8	9	10	11	12	13	14
Lena	1620	755	871	1740	534	532	365
Baboon	1061	397	610	1116	320	381	290
Category	15	16	17	18	19	20	21
Lena	337	255	500	423	375	284	308
Baboon	482	306	399	338	315	300	367
Category	22	23	24	25	26	27	28
Lena	285	254	225	221	257	220	210
Baboon	331	373	304	275	211	199	243

3.2. **Proposed feature.** In this section, the proposed method is elaborated by introducing how to derive an image feature descriptor from the BTC data stream and the bit-plane image. Different from most of the BTC and the BTC-VQ based feature extraction schemes, we add the edge orientation classification of bit-plane that can extract more edge orientation information of image blocks. In general, our proposed feature is the combination of the CCHF and the EOBP classification based features.

3.2.1. *Contrast and Color Histogram based Feature (CCHF).* The Contrast and Color Histogram based Feature (CCHF), are derived from the color distribution, image contrast [5]. The color distribution of images contains large amount of information about the image, and it is the most intuitive expression of the image. In our proposed scheme, the $M \times N$ YCbCr images are firstly divided into non-overlapping 4 × 4 sub-blocks, i.e., $\mathbf{X} = \{x_{ij} | 1 \leq i \leq M/4, 1 \leq j \leq N/4\}$. And the image blocks are encoded with the AMBTC [23] based compression. And we will get two mean level and 4 × 4 sized bit-plane $\boldsymbol{bp}_{ij}$. The two mean levels are high and low mean levels: h_{ij} and l_{ij}. Then we design the codebooks for h_{ij} and l_{ij} as C_{high} and C_{low} by performing the TSVQ [25] algorithm.

After the codebooks are obtained, all blocks are encoded with the predesigned codebook consisting of N codewords to obtain an index sequence, and then calculate the histogram of the color indexing and bit pattern indexing from the sequence, and finally we will obtain the co-occurrence feature and the bit pattern feature, named as CCHF.

Next we contrast the high and low quantizer. Broadly speaking, the CCHF are the TS-VQ-indexed histogram from the color high and low quantizers, respectively. Figure 5 illustrates the schematic diagram for computing the CCHF.

3.2.2. *Edge Orientation of Bit-Plane Feature (EOBPF).* As is known, if BTC is applied to preserving the locations of the light and dark pixels inside the edge block then TSVQ [25] is applied to encoding the light and dark pixels independently.

In order to expect the better edge reproduction of the bit-plane image, we divided the bit-plane image into more elaborate categories according to the EOBP templates. Moreover, each of the categories represents different edge orientations of the bit-plane. After the EOBP templates are performed on the bit-plane image, some blocks in the image are encoded with index i. Then we count the number of blocks belonging to each category with each index i, obtaining its class index histogram, which is called as EOBPF. The block diagram of the EOBPF based scheme is shown in Figure 6.

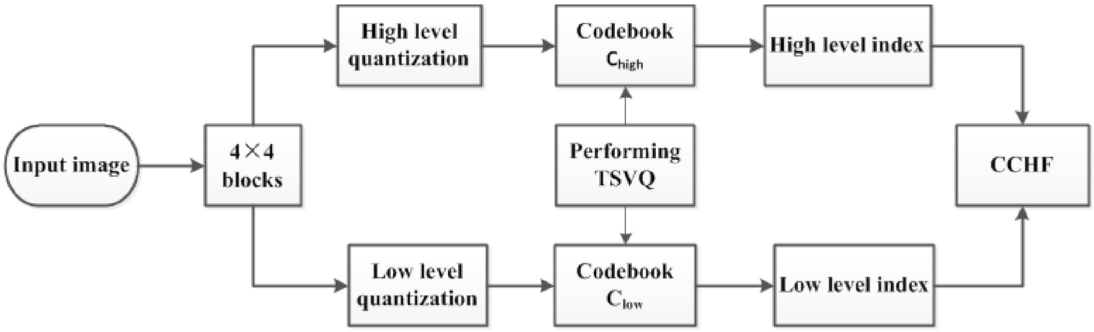


Figure 5. Block diagram for computing the CCHF

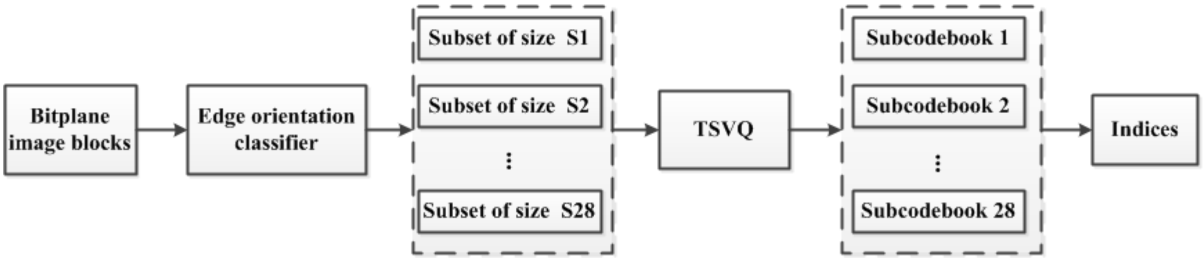


Figure 6. Block diagram for computing the EOBPF

Table 2. Image database employed in this study

Database Name	Image Size	Number of Classes	Images in Each Class	Database Size
Corel 1,000	384 × 256	10	100	1000
Corel 10,000	384 × 256	100	100	10000

4. **Simulations Results and Discussion.** We perform simulations on various types of datasets listed in Table 2 which are examined in this study, including Corel 1,000, Corel 10,000 [18]. And the detailed information for each database is listed in Table 2.

For example, the Corel 1,000 database is classified into ten classes, each class including 100 images, named as Class1_people, Class2_beach, Class3_buliding, Class4_bus, Class5_dinosaur, Class6_elephant, Class7_flower, Class8_horse, Class9_mountain and Class10_food respectively. Similarly, the 100 categories of Corel 10,000 are also natural images.

4.1. **Proposed method for image compression.** Firstly, we do experiments to test the compression ratio of our proposed edge orientation classification of bit-plane templates. Here, to demonstrate the efficiency of the proposed templates based coding, we take the Peak Signal to Noise Ratio (PSNR) [19] as our evaluation parameter. For example, for an image of $M \times N$ with the gray-level of L, we define its original pixel as x_{ij} and its reconstructed image pixels as y_{ij}, $0 \le i \le M-1$, $0 \le j \le N-1$, PSNR (dB) can be defined as follows:

$$\mathrm{MSE} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(x_{ij}-y_{ij}\right)^2}{M \times N} \qquad \mathrm{PSNR} = 10 \times \log_{10}\frac{(L-1)^2}{\mathrm{MSE}} \tag{10}$$

where MSE denotes the Mean Squared Error.

As the traditional BTC-VQ based compression can combine the advantage of BTC and VQ, while the edge orientation of the bit-plane image is not taken into account. Our proposed templates can reserve more edge orientation information, and we propose a new image compression method combining the EOBP classification with the TSVQ, EOBP-VQ for short. Here we do experiments and compare our EOBP-VQ with the traditional BTC-VQ.

In our experiments, for example, we do experiments on the database of 1000 images and obtain the PSNR (dB) values of Y, Cb, Cr and the average PSNR (dB) values of the three components for each image. The average results are shown in Table 3.

Table 3. The average PSNR (dB) based on codebook size of 256 for based compression

Image	BTC-VQ				EOBP-TSVQ			
Database	Y	Cb	Cr	Average	Y	Cb	Cr	Average
Lena	29.9726	38.6678	39.7582	36.1329	30.3367	39.5484	40.8082	36.89777
Baboon	22.9089	33.1582	34.9885	30.3519	25.4662	34.5544	35.6627	31.89443
Corel 1,000	26.7342	42.7825	42.6753	37.3973	28.9270	43.5671	43.6721	38.7221
Corel 10,000	28.8125	43.6787	43.0238	38.5050	28.9889	44.7654	43.9793	39.2445

From the results in Table 3, we can easily get three conclusions: (1) For each type of image, with the same size of codebook, our proposed EOBP-TSVQ based image compression performs better than the BTC-VQ based compression for the components Y, Cb and Cr; (2) As we calculate the average PSNR (dB) values according to $(\mathrm{Y}+\mathrm{Cb}+\mathrm{Cr})/3$, our proposed scheme also has a better performance than the BTC-VQ based scheme; (3) Comparatively speaking, the component Y plays a more important role than components Cb and Cr for compression performance.

4.2. **Proposed feature for image retrieval.**

Secondly, we perform the EOBP templates on the bit-plane image, and the proposed feature is the combination of the EOBP based feature with the CCHF. In order to verify the effectiveness of the proposed scheme, we compare our feature with the VQIH, BTCH and the BTC-VQIH features. The comparison results are shown in Figure 7. To compare the performance more reasonably, we do experiments on the Corel 1,000 of 1,000 images and the Corel 10,000 of 10,000 images. Firstly, we randomly select five images from each class, so that a total of 50 images are used as test query images. For each test query image, we perform the retrieval process based on each kind of features. For each number of returned images (from 1 to 1,000 or 1 to 10,000), we calculate the average recall and precision value over 50 test query images. Here, the precision and recall are defined as follows:

$$\text{precision} = \frac{\text{No. relevant images}}{\text{No. images returned}} \qquad \text{recall} = \frac{\text{No. relevant images}}{\text{No. total images}} \tag{11}$$

Figure 7 compares the average P-R curves among VQIH [21], BTCH [4], BTC-VQIH [5] and our proposed feature; we can easily find that our proposed feature can obtain a much better performance in precision and recall. The main reason is that our feature consists of the color co-occurrence feature and bit pattern feature as well as the edge information. Our EOBP based feature can reflect the edge orientation information. In addition, the CCHF can characterize the color feature of the image. On the other hand, the CCHF is also compressed by TSVQ, and the corresponding index histograms can reflect the texture information. To sum up, compared with the BTC and BTC-VQ based features, our feature can capture more edge information of each block. Therefore, our scheme is superior to the traditional VQIH-, BTC- and BTC-VQ-based schemes.

To see the performance of our proposed algorithm more directly, we give a further comparison. Here, we show the example of retrieved images from the natural image

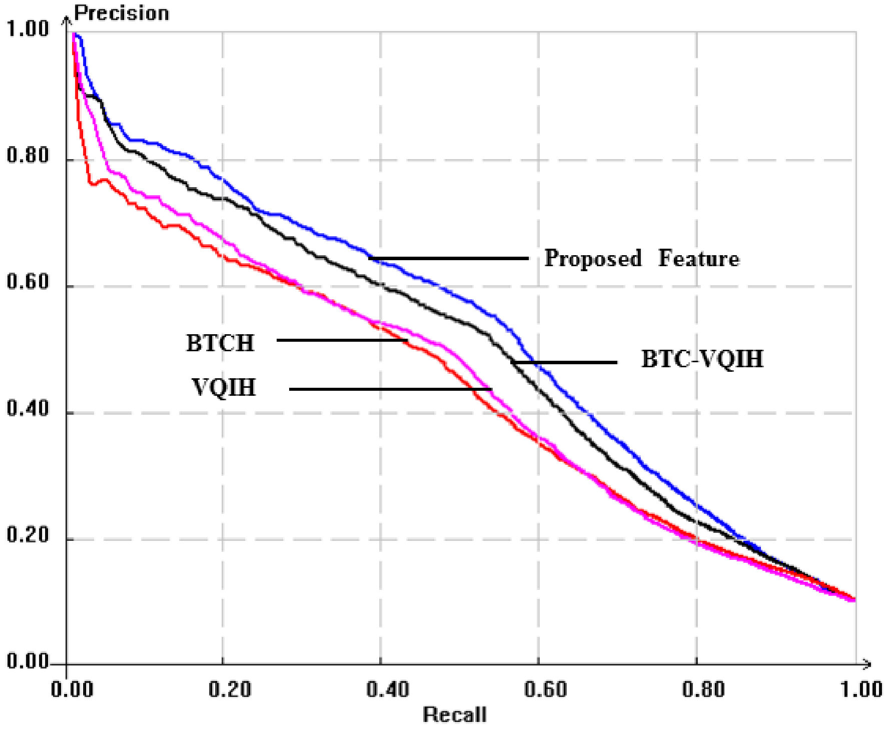
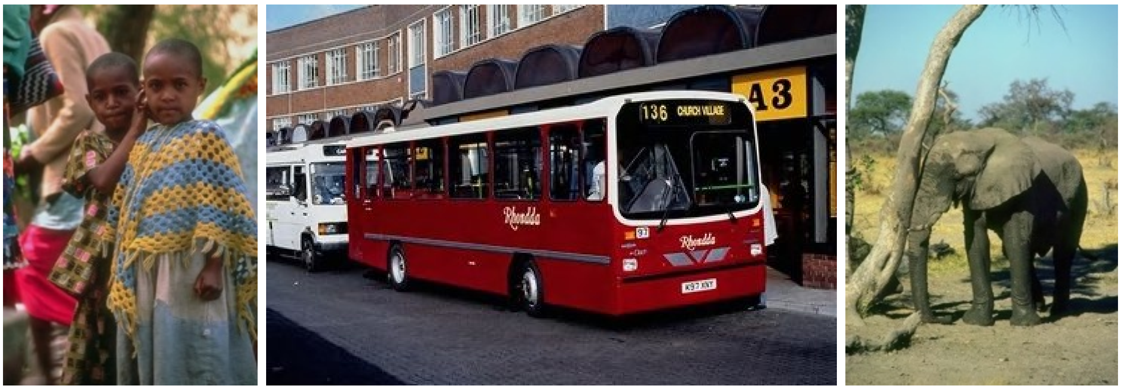


Figure 7. The comparisons of the average P-R curves of VQIH, BTCH, BTC-VQIH and proposed features for Corel 10,000 database



(a) (b) (c)

Figure 8. The examples of query images

database Corel 1,000 for BTC-VQ and the proposed feature. Take three of the images from the Class1_people, Class4_bus and, Class6_elephant as an examples as shown in Figure 8.

All of the images in the database are sorted according to the order from the most similar one to the least similar one. We show only the first 15 retrieved images with the highest similarities due to the limitation of the space and rank the distance (Dis.1 $\sim$ Dis.15) between the query image and the retrieved images from low to high in Tables 4-6.

From the comparison results in Tables 4-6, the most similar image is the query image itself, so Dis.1 $=$ 0; and distances (Dis.2 $\sim$ Dis.15) increase with the difference between the query image and the retrieved images. In addition, for the three different images, distances (dis) in our scheme are much lower than the other three schemes; that is to say, it is more possible for our scheme to retrieve similar images. From the point of practical application, our scheme is superior to the other three typical schemes. Furthermore, the distance varies with different categories of images, so it can be a good feature to perform image classification task.

As the BTC-VQ based scheme is the combination of BTCH and VQIH based schemes, it takes advantages of both the BTCH and VQIH. Here, we just show the retrieval results of BTC-VQIH and the proposed schemes in Figures 9-11.

From Figures 9-11, we can see that our scheme performs better than the BTC-VQIH based scheme. For example, in Figures 9 and Figure 11, the proposed scheme can retrieve more right images than the BTC-VQ based scheme. In Figure 10, both schemes can retrieve 16 images, while what is deserved to be mentioned is that the distance (dis)

Table 4. Similarity distance comparison for (a) in Figure 8

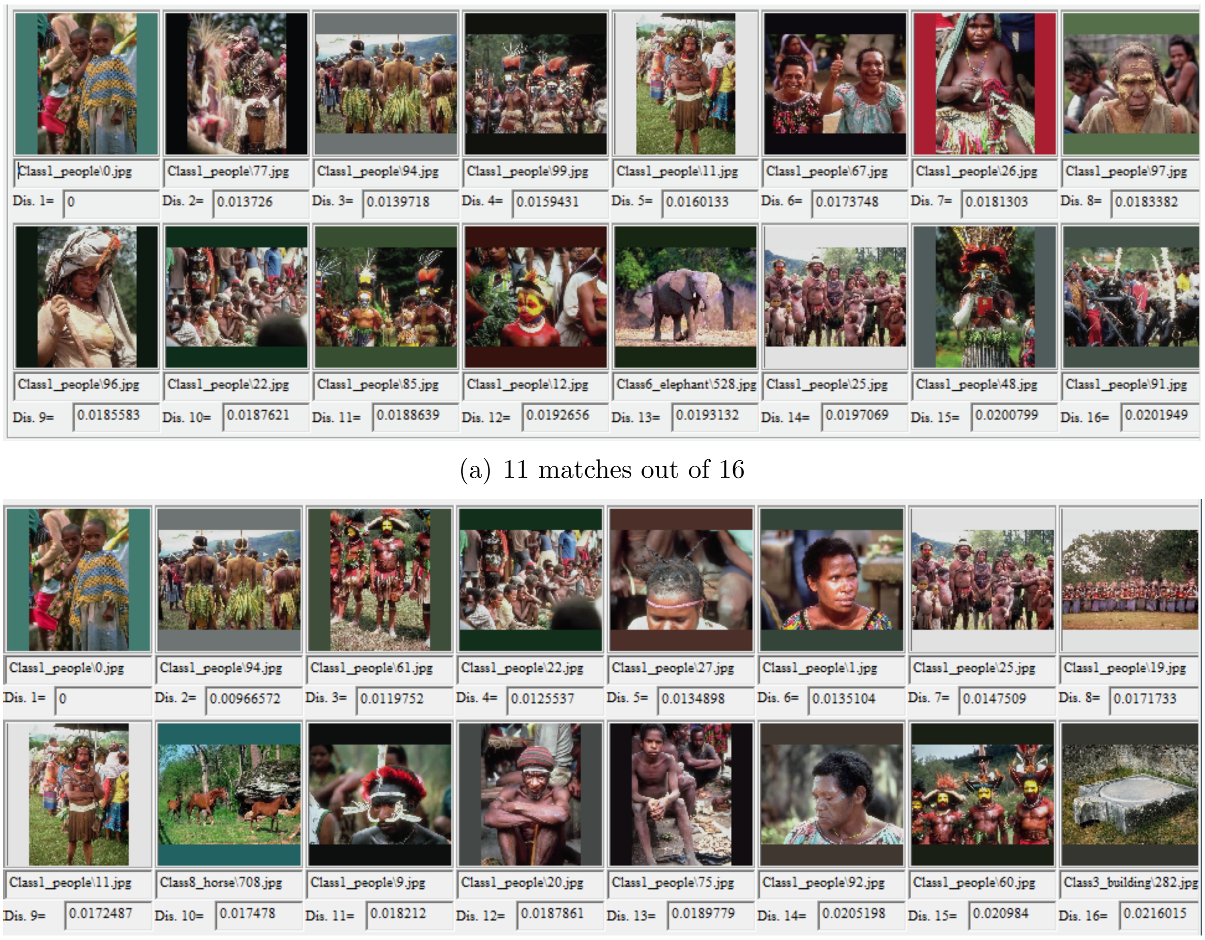
Distance	BTCH	VQIH	BTC-VQIH	Proposed
Dis.1	0	0	0	0
Dis.2	0.0741325	0.0585787	0.013726	0.0096657
Dis.3	0.0782966	0.0759199	0.0139718	0.0119752
Dis.4	0.0807419	0.0762461	0.0159431	0.0125537
Dis.5	0.0828025	0.0801185	0.0160133	0.0134898
Dis.6	0.0878468	0.0832525	0.0173748	0.0135104
Dis.7	0.0886366	0.0834545	0.0181303	0.0147509
Dis.8	0.0887486	0.0850039	0.0183382	0.0171533
Dis.9	0.0893977	0.088243	0.0185583	0.0172487
Dis.10	0.0894723	0.0887047	0.0187621	0.017478
Dis.11	0.0907191	0.0892978	0.0188639	0.018212
Dis.12	0.0910483	0.0917666	0.0192656	0.0187681
Dis.13	0.0926473	0.0935643	0.0193132	0.0189779
Dis.14	0.0948718	0.0935997	0.0197069	0.0205198
Dis.15	0.0956424	0.095036	0.0200799	0.020984

Table 5. Similarity distance comparison for (b) in Figure 8

Distance	BTCH	VQIH	BTC-VQIH	Proposed
Dis.1	0	0	0	0
Dis.2	0.0322261	0.0567749	0.0890749	0.0714058
Dis.3	0.0374406	0.0641566	0.0979896	0.0794718
Dis.4	0.0403317	0.0658186	0.100265	0.0812209
Dis.5	0.0407286	0.067093	0.100441	0.0819461
Dis.6	0.0435425	0.0690529	0.101455	0.0847586
Dis.7	0.0444224	0.069067	0.102671	0.0848869
Dis.8	0.044943	0.0700279	0.103758	0.0904816
Dis.9	0.0450939	0.0708618	0.106394	0.0912825
Dis.10	0.0459912	0.0709968	0.107709	0.0920636
Dis.11	0.0460424	0.0715103	0.109328	0.0945597
Dis.12	0.0463576	0.0723534	0.109837	0.0949798
Dis.13	0.047015	0.073142	0.114205	0.0951262
Dis.14	0.0477249	0.0745921	0.11495	0.097819
Dis.15	0.047746	0.0748283	0.115098	0.0991044

Table 6. Similarity distance comparison for (c) in Figure 8

Distance	BTCH	VQIH	BTC-VQIH	Proposed
Dis.1	0	0	0	0
Dis.2	0.0338169	0.0372294	0.0321436	0.0154924
Dis.3	0.057532	0.0490489	0.0414913	0.0232357
Dis.4	0.0601753	0.0493317	0.04365	0.0254836
Dis.5	0.0643589	0.0528338	0.0526857	0.0261624
Dis.6	0.0643728	0.0554024	0.0531216	0.0261839
Dis.7	0.0643877	0.0595427	0.057129	0.0278694
Dis.8	0.0654174	0.0598403	0.0586309	0.0286031
Dis.9	0.0658439	0.0606164	0.0604894	0.0289644
Dis.10	0.0668272	0.0614572	0.0613118	0.0290958
Dis.11	0.0668592	0.0626531	0.06165	0.0294833
Dis.12	0.0676368	0.0628476	0.0623392	0.0295665
Dis.13	0.0683398	0.0630668	0.0630026	0.029934
Dis.14	0.069079	0.0634736	0.0633639	0.0305882
Dis.15	0.0696514	0.0635595	0.0634727	0.0308549



Class1_people\0.jpg
Dis. 1= 0

Class1_people\77.jpg
Dis. 2= 0.013726

Class1_people\94.jpg
Dis. 3= 0.0139718

Class1_people\99.jpg
Dis. 4= 0.0159431

Class1_people\11.jpg
Dis. 5= 0.0160133

Class1_people\67.jpg
Dis. 6= 0.0173748

Class1_people\26.jpg
Dis. 7= 0.0181303

Class1_people\97.jpg
Dis. 8= 0.0183382

Class1_people\96.jpg
Dis. 9= 0.0185583

Class1_people\22.jpg
Dis. 10= 0.0187621

Class1_people\85.jpg
Dis. 11= 0.0188639

Class1_people\12.jpg
Dis. 12= 0.0192656

Class6_elephant\528.jpg
Dis. 13= 0.0193132

Class1_people\25.jpg
Dis. 14= 0.0197069

Class1_people\48.jpg
Dis. 15= 0.0200799

Class1_people\91.jpg
Dis. 16= 0.0201949

(a) 11 matches out of 16

Class1_people\0.jpg
Dis. 1= 0

Class1_people\94.jpg
Dis. 2= 0.00966572

Class1_people\61.jpg
Dis. 3= 0.0119752

Class1_people\22.jpg
Dis. 4= 0.0125537

Class1_people\27.jpg
Dis. 5= 0.0134898

Class1_people\1.jpg
Dis. 6= 0.0135104

Class1_people\25.jpg
Dis. 7= 0.0147509

Class1_people\19.jpg
Dis. 8= 0.0171733

Class1_people\11.jpg
Dis. 9= 0.0172487

Class8_horse\708.jpg
Dis. 10= 0.017478

Class1_people\9.jpg
Dis. 11= 0.018212

Class1_people\20.jpg
Dis. 12= 0.0187861

Class1_people\75.jpg
Dis. 13= 0.0189779

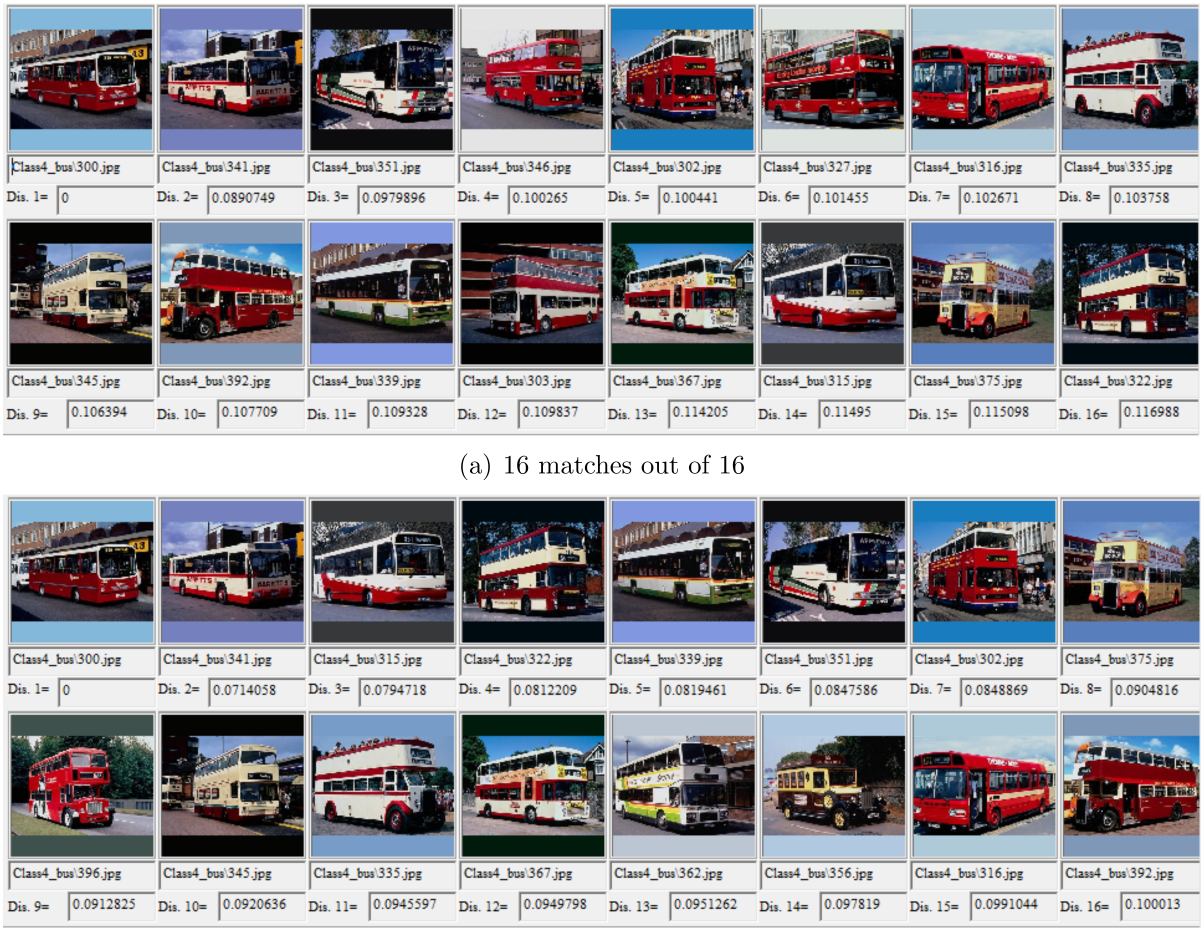
Class1_people\92.jpg
Dis. 14= 0.0205198

Class1_people\60.jpg
Dis. 15= 0.020984

Class3_building\282.jpg
Dis. 16= 0.0216015

(b) 14 matches out of 16

Figure 9. Retrieval results with the two schemes. The query image belongs to "Class1". (a) BTC-VQIH, (b) proposed feature.



Class4_bus\300.jpg	Class4_bus\341.jpg	Class4_bus\351.jpg	Class4_bus\346.jpg	Class4_bus\302.jpg	Class4_bus\327.jpg	Class4_bus\316.jpg	Class4_bus\335.jpg
Dis. 1= 0	Dis. 2= 0.0890749	Dis. 3= 0.0979896	Dis. 4= 0.100265	Dis. 5= 0.100441	Dis. 6= 0.101455	Dis. 7= 0.102671	Dis. 8= 0.103758
Class4_bus\345.jpg	Class4_bus\392.jpg	Class4_bus\339.jpg	Class4_bus\303.jpg	Class4_bus\367.jpg	Class4_bus\315.jpg	Class4_bus\375.jpg	Class4_bus\322.jpg
Dis. 9= 0.106394	Dis. 10= 0.107709	Dis. 11= 0.109328	Dis. 12= 0.109837	Dis. 13= 0.114205	Dis. 14= 0.11495	Dis. 15= 0.115098	Dis. 16= 0.116988

(a) 16 matches out of 16

Class4_bus\300.jpg	Class4_bus\341.jpg	Class4_bus\315.jpg	Class4_bus\322.jpg	Class4_bus\339.jpg	Class4_bus\351.jpg	Class4_bus\302.jpg	Class4_bus\375.jpg
Dis. 1= 0	Dis. 2= 0.0714058	Dis. 3= 0.0794718	Dis. 4= 0.0812209	Dis. 5= 0.0819461	Dis. 6= 0.0847586	Dis. 7= 0.0848869	Dis. 8= 0.0904816
Class4_bus\396.jpg	Class4_bus\345.jpg	Class4_bus\335.jpg	Class4_bus\367.jpg	Class4_bus\362.jpg	Class4_bus\356.jpg	Class4_bus\316.jpg	Class4_bus\392.jpg
Dis. 9= 0.0912825	Dis. 10= 0.0920636	Dis. 11= 0.0945597	Dis. 12= 0.0949798	Dis. 13= 0.0951262	Dis. 14= 0.097819	Dis. 15= 0.0991044	Dis. 16= 0.100013

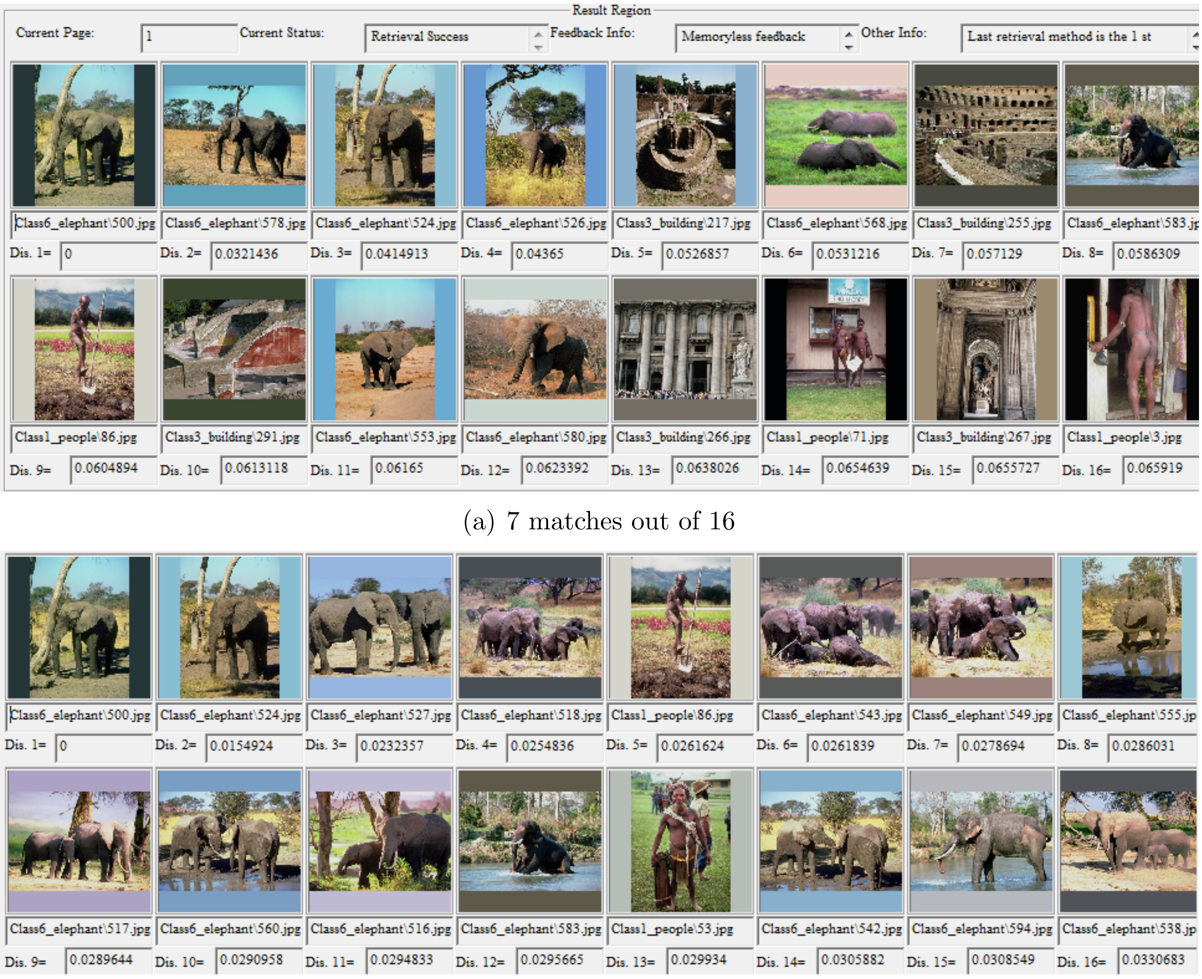
(b) 16 matches out of 16

Figure 10. Retrieval results with the two schemes. The query image belongs to "Class4". (a) BTC-VQIH, (b) proposed feature.

between the query image and the retrieved images is much lower in our scheme, that is to say, it is more possible for our scheme to retrieve similar images. From the point of practical application, our scheme is superior to the BTC-VQ based scheme. Thus, we can conclude that the recall and precision values of our scheme are both higher than those of the BTC-VQ based image retrieval scheme for the three categories of images.

Figure 12 shows the comparison results of precision between the proposed method and former schemes using Corel 1,000 and Corel 10,000 image databases. It can be seen that the proposed method achieves better performance than other listed methods for Corel 1,000 and Corel 10,000. Compared with other BTC based feature, our feature takes the advantages of both BTC and VQ, and it also overcomes the edge ladder effect. Therefore, our edge-preserving algorithm can be an efficient feature for natural image database retrieval.

5. **Conclusions and Future Directions.** In this study, an image retrieval scheme is proposed by exploiting the EOBP templates classifying the bit-plane image into 28-categories. And we employ the EOBP classification on bit-plane image blocks and do experiments for image compression and image retrieval. In terms of compression ratio, the proposed EOBP-TSVQ based algorithm is superior to the BTC-VQ based algorithm. As for image retrieval, our feature is extracted by combining the proposed EOBP based



Result Region

Current Page: 1 Current Status: Retrieval Success Feedback Info: Memoryless feedback Other Info: Last retrieval method is the 1 st

Class6_elephant\500.jpg Dis. 1= 0
Class6_elephant\578.jpg Dis. 2= 0.0321436
Class6_elephant\524.jpg Dis. 3= 0.0414913
Class6_elephant\526.jpg Dis. 4= 0.04365
Class3_building\217.jpg Dis. 5= 0.0526857
Class6_elephant\568.jpg Dis. 6= 0.0531216
Class3_building\255.jpg Dis. 7= 0.057129
Class6_elephant\583.jp Dis. 8= 0.0586309
Class1_people\86.jpg Dis. 9= 0.0604894
Class3_building\291.jpg Dis. 10= 0.0613118
Class6_elephant\553.jpg Dis. 11= 0.06165
Class6_elephant\580.jpg Dis. 12= 0.0623392
Class3_building\266.jpg Dis. 13= 0.0638026
Class1_people\71.jpg Dis. 14= 0.0654639
Class3_building\267.jpg Dis. 15= 0.0655727
Class1_people\3.jpg Dis. 16= 0.065919

(a) 7 matches out of 16

Class6_elephant\500.jpg Dis. 1= 0
Class6_elephant\524.jpg Dis. 2= 0.0154924
Class6_elephant\527.jpg Dis. 3= 0.0232357
Class6_elephant\518.jpg Dis. 4= 0.0254836
Class1_people\86.jpg Dis. 5= 0.0261624
Class6_elephant\543.jpg Dis. 6= 0.0261839
Class6_elephant\549.jpg Dis. 7= 0.0278694
Class6_elephant\555.jp Dis. 8= 0.0286031
Class6_elephant\517.jpg Dis. 9= 0.0289644
Class6_elephant\560.jpg Dis. 10= 0.0290958
Class6_elephant\516.jpg Dis. 11= 0.0294833
Class6_elephant\583.jpg Dis. 12= 0.0295665
Class1_people\53.jpg Dis. 13= 0.029934
Class6_elephant\542.jpg Dis. 14= 0.0305882
Class6_elephant\594.jpg Dis. 15= 0.0308549
Class6_elephant\538.jp Dis. 16= 0.0330683

(b) 14 matches out of 16

FIGURE 11. Retrieval results with the two schemes. The query image belongs to "Class6". (a) BTC-VQIH, (b) proposed feature.

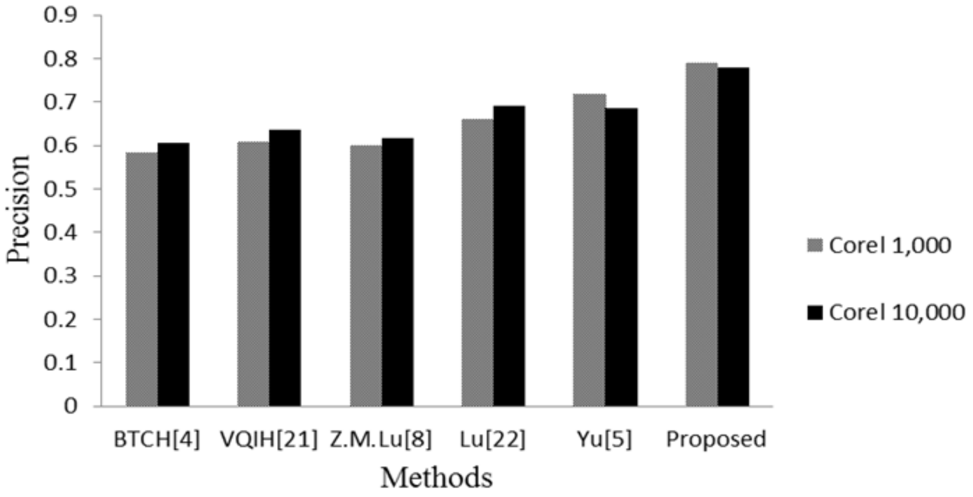


FIGURE 12. Comparison with former existing methods for Corel 1,000 and Corel 10,000 image databases

feature and the CCHF. As documented in the experimental results, the proposed scheme can provide better average precision and recall compared to various former schemes in the literature. Therefore, the proposed scheme can be considered as a very competitive candidate in color image retrieval application.

In the future studies, we will concentrate on applying the proposed hybrid feature to other application areas, such as image identification, copyright protection and content authentication.

REFERENCES

[1] W. J. Han and K. A. Sohn, Image classification approach for improving CBIR system performance, *The Journal of Korean Institute of Communications and Information Sciences*, vol.41, no.7, pp.816-822, 2016.

[2] E. Delp and O. Mitchell, Image compression using block truncation coding, *IEEE Trans. Communications*, vol.27, no.9, pp.1335-1342, 1979.

[3] R. Gray, Vector quantization, *IEEE ASSP Magazine*, vol.1, no.2, pp.4-29, 1984.

[4] G. Qiu, Color image indexing using BTC, *IEEE Trans. Image Processing*, vol.12, no.1, pp.93-101, 2003.

[5] F. X. Yu, H. Luo and Z. M. Lu, Color image retrieval using pattern co-occurrence matrices based on BTC and VQ, *Electron Letters*, vol.47, no.2, pp.100-101, 2011.

[6] J. M. Guo, H. Prasetyo and C. C. Yao, Content-based image retrieval using direct binary search block truncation coding features, *Asia-Pacific Signal and Information Processing Association Summit and Conference*, pp.693-696, 2015.

[7] J. M. Guo, H. Prasetyo and J. H. Chen, Content-based image retrieval using error diffusion block truncation coding features, *IEEE Trans. Circuits and Systems for Video Technology*, vol.25, no.3, pp.466-481, 2015.

[8] Z. M. Lu and H. Burkhardt, Color image retrieval based on DCT-domain vector quantization index histograms, *Electronics Letters*, vol.41, no.17, pp.956-957, 2005.

[9] Z. M. Lu and Y. P. Feng, Image retrieval based on histograms of EOPs and VQ indices, *Electronics Letters*, vol.52, no.20, pp.1683-1684, 2016.

[10] L. Wang, Z. M. Lu, L. H. Ma and Y. P. Feng, VQ codebook design using modified K-means algorithm with feature classification and grouping based initialization, *Multimedia Tools and Applications*, vol.10, pp.1-16, 2017.

[11] F. D. V. R. Oliveira, H. L. Haas, J. G. R. C. Gomes and A. Petraglia, CMOS imager with focal-plane analog image compression combining DPCM and VQ, *IEEE Trans. Circuits and Systems I: Regular Papers*, vol.60, no.5, pp.1331-1344, 2013.

[12] C. Qin and Y. C. Hu, Reversible data hiding in VQ index table with lossless coding and adaptive switching mechanism, *Signal Processing*, vol.129, no.1, pp.48-55, 2016.

[13] Y. W. P. Hong, L. M. Huang and H. T. Li, Vector quantization and clustered key mapping for channel-based secret key generation, *IEEE Trans. Information Forensics and Security*, vol.12, no.5, pp.1170-1181, 2017.

[14] W. Zhao and L. Lai, Distributed detection with vector quantizer, *IEEE Trans. Signal and Information Processing over Networks*, vol.2, no.2, pp.105-119, 2016.

[15] H. B. Kekre, S. D. Thepade, T. K. Sarode and N. Bhandari, Colorization of grayscale images using LBG VQ codebook for different color spaces, *International Journal of Engineering Research and Applications (IJERA)*, vol.1, no.4, pp.1274-1283, 2014.

[16] S. Z. Su, S. Y. Chen and S. Z. Li, Structured local binary Haar pattern for pixel-based graphics retrieval, *Electronics Letters*, vol.46, no.14, pp.996-998, 2010.

[17] G. Y. Kang, S. Z. Guo, D. C. Wang, L. H. Ma and Z. M. Lu, Image retrieval based on structured local binary Kirsch pattern, *Trans. Inf. Syst*, vol.96, no.5, pp.1230-1232, 2013.

[18] *Corel Photo Collection Color Image Database*, http://wang.ist.psu.edu/docs/realted/.

[19] P. Kaushik and Y. Sharma, Comparison of different image enhancement techniques based upon PSNR & MSE, *International Journal of Applied Engineering Research*, vol.7, no.11, pp.2010-2014, 2012.

[20] H. Dujmic, N. Rozic and D. Begusic, Local thresholding classified vector quantization with memory reduction, *International Workshop on Image and Signal Processing and Analysis*, pp.197-202, 2000.

[21] S. W. Teng and G. Lu, Image indexing and retrieval based on vector quantization, *Pattern Recognition*, vol.40, no.11, pp.3299-3316, 2007.

[22] T. C. Lu and C. C. Chang, Color image retrieval technique based on color features and image bitmap, *Information Processing & Management*, vol.43, no.2, pp.461-472, 2007.

[23] W. Hong, T. S. Chen and Z. Yin, Data hiding in AMBTC images using quantization level modification and perturbation technique, *Multimedia Tools & Applications*, vol.76, no.3, pp.3761-3782, 2017.

[24] J. M. Guo and Y. F. Liu, Improved block truncation coding using optimized dot diffusion, *IEEE Trans. Image Processing*, vol.23, no.3, pp.1269-1275, 2014.

[25] H. Wu, Q. Wang and M. Flierl, Tree-structured vector quantization for similarity queries, *Data Compression Conference*, p.467, 2017.